

**Amendments to the Claims:**

The following listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Previously Presented) A method to be executed by a computer system for determining whether a computer-storable expression matches a filter, the method comprising:  
  
evaluating a first code structure representing the expression for determining a value of said expression;  
  
analyzing a second code structure representing the filter for determining the characteristics of the filter; and  
  
filtering said value according to the filter characteristics,  
  
wherein said first code structure is constructed from a plurality of first programming language code structure elements and said second code structure is constructed from a plurality of second programming language code structure elements; each second structure element corresponding to one of said first structure elements, and  
  
wherein evaluating, analyzing and filtering are performed upon explicit invocation of a matching operator, and filtering comprises returning a boolean evaluation result value.
2. (Canceled)
3. (Previously Presented) The method of claim 1, wherein the second code structure includes at least one composition operator acting as a logical connector for logically combining two of said programming language code structure elements, or for inverting the boolean value of at least one of said second programming language code structure elements.
4. (Original) The method of claim 1, wherein the first code structure includes a first concatenation operator for concatenating two expressions, and the second code structure

includes a second concatenation operator for concatenating two filter elements, the first and the second concatenation operators being applied within the first code structure and the second code structure, respectively, in essentially the same manner.

5. (Original) The method of claim 1, wherein the first and the second code structures include indicator elements indicating a data type, the indicator elements acting as structure constructors in the first code structure and as filter constructors in the second code structure, each of the structure constructors corresponding to a respective one of the filter constructors.

6. (Previously Presented) The method of claim 1, wherein the second code structure includes a test operator having an operand, and wherein filtering comprises testing the occurrence of the value of said operand in the expression.

7. (Original) The method of claim 1, wherein the second code structure includes an existence operator that matching any element that exists.

8. (Original) The method of claim 1, wherein the second code structure includes an assignment operator having an operand, to assign a part of the expression to a variable that is identified by said operand.

9. (Original) The method of claim 1, wherein the second code structure includes a Kleene operator.

10. (Previously Presented) The method of claim 1, wherein the second code structure includes a do operator having two arguments, one argument being a filter and the other argument being an instruction or a sequence of instructions, wherein filtering includes executing the instructions only if the filter is successful.

11. (Original) The method of claim 1, wherein the filter is a recursive filter enabling filtering of trees.

12. (Original) The method of claim 1, wherein the filter is a normalized filter.
13. (Previously Presented) The method of claim 1, wherein filtering includes modifying the environment of the computer system, the environment including variables and corresponding values used by the computer system when filtering said value of said expression.
14. (Original) The method of claim 1, wherein the first code structure and the second code structure are part of an interpreter programming language code.
15. (Previously Presented) The method of claim 1, wherein the first code structure and the second code structure are part of a compiler programming language code.
- 16-18. (Canceled)